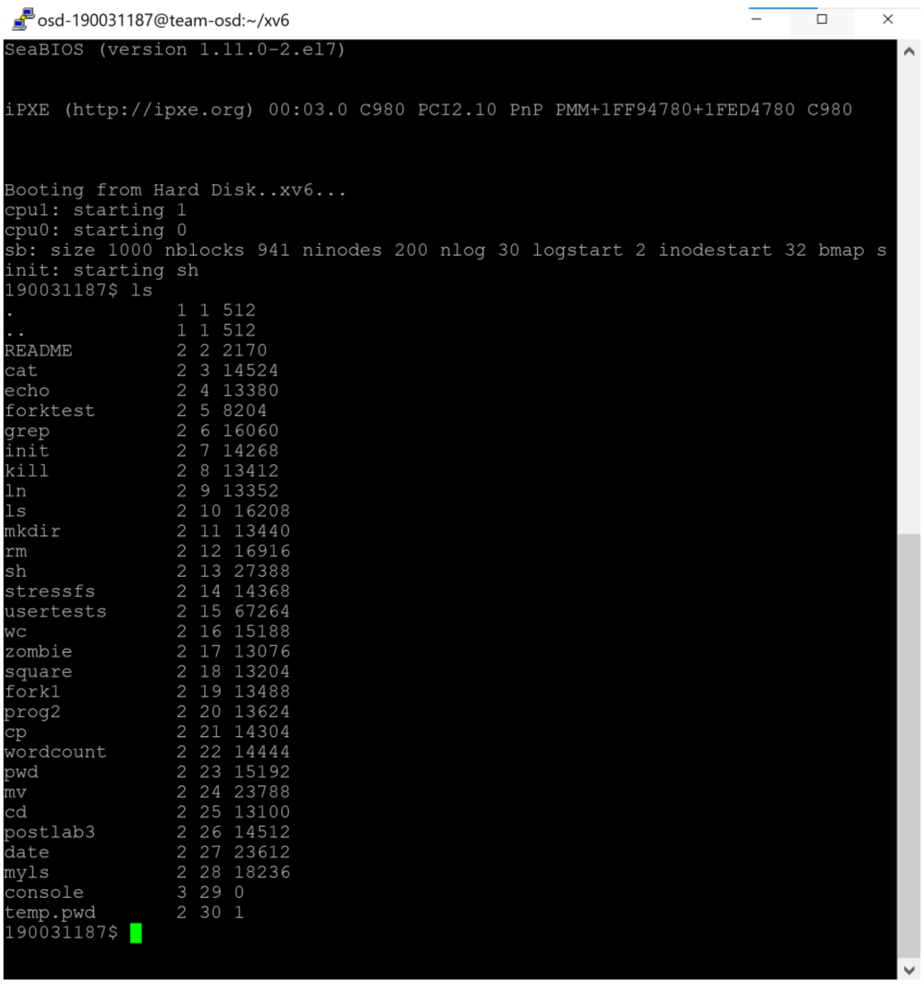
Operating System and Design (19CS2106S)
Skill - 5
1) ls.c (Xv6 design & implementation. (xv6 source code))

```
#include "types.h"
#include "stat.h"
#include "user.h"
#include "fs.h"
char* fmtname(char *path)
{
  static char buf[DIRSIZ+1];
  char *p;
  // Find first character after last slash.
  for(p=path+strlen(path); p >= path && *p != '/'; p--);
  p++;
  // Return blank-padded name.
  if(strlen(p) >= DIRSIZ)
      return p;
  memmove(buf, p, strlen(p));
  memset(buf+strlen(p), ' ', DIRSIZ-strlen(p));
  return buf;
}
void ls(char *path)
{
  char buf[512], *p;
  int fd;
  struct dirent de;
  struct stat st;
  if((fd = open(path, 0)) < 0){
    printf(2, "ls: cannot open %s\n", path);
  return;
}
if(fstat(fd, &st) < 0){
    printf(2, "ls: cannot stat %s\n", path);
    close(fd);
    return;
}
switch(st.type){
case T_FILE:
            printf(1, "%s %d %d %d\n", fmtname(path),   st.type, st.ino, st.size);
            break;
case T_DIR:
            if(strlen(path) + 1 + DIRSIZ + 1 > sizeof buf){
            printf(1, "ls: path too long\n");
            break;
            }
            strcpy(buf, path);
            p = buf+strlen(buf);
            *p++ = '/';
while(read(fd, &de, sizeof(de)) == sizeof(de)){
            if(de.inum == 0 ) continue;
```

```
        memmove(p, de.name, DIRSIZ);
        p[DIRSIZ] = 0;
        if(stat(buf, &st) < 0){
                printf(1, "ls: cannot stat %s\n", buf);
                continue;
        }
        printf(1, "%s %d %d %d\n", fmtname(buf), st.type, st.ino, st.size);
}
break;
}
close(fd);
}
Int   main(int argc, char *argv[])
{
        int i;
        if(argc < 2){
                ls(".");
                exit();
        }
        for(i=1; i<argc; i++)
                ls(argv[i]);
        exit();
}
```

OUTPUT



osd-190031187@team-osd:~/xv6

```
SeaBIOS (version 1.11.0-2.el7)


iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980



Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap s
init: starting sh
190031187$ ls
.              1 1 512
..             1 1 512
README         2 2 2170
cat            2 3 14524
echo           2 4 13380
forktest       2 5 8204
grep           2 6 16060
init           2 7 14268
kill           2 8 13412
ln             2 9 13352
ls             2 10 16208
mkdir          2 11 13440
rm             2 12 16916
sh             2 13 27388
stressfs       2 14 14368
usertests      2 15 67264
wc             2 16 15188
zombie         2 17 13076
square         2 18 13204
fork1          2 19 13488
prog2          2 20 13624
cp             2 21 14304
wordcount      2 22 14444
pwd            2 23 15192
mv             2 24 23788
cd             2 25 13100
postlab3       2 26 14512
date           2 27 23612
myls           2 28 18236
console        3 29 0
temp.pwd       2 30 1
190031187$
```

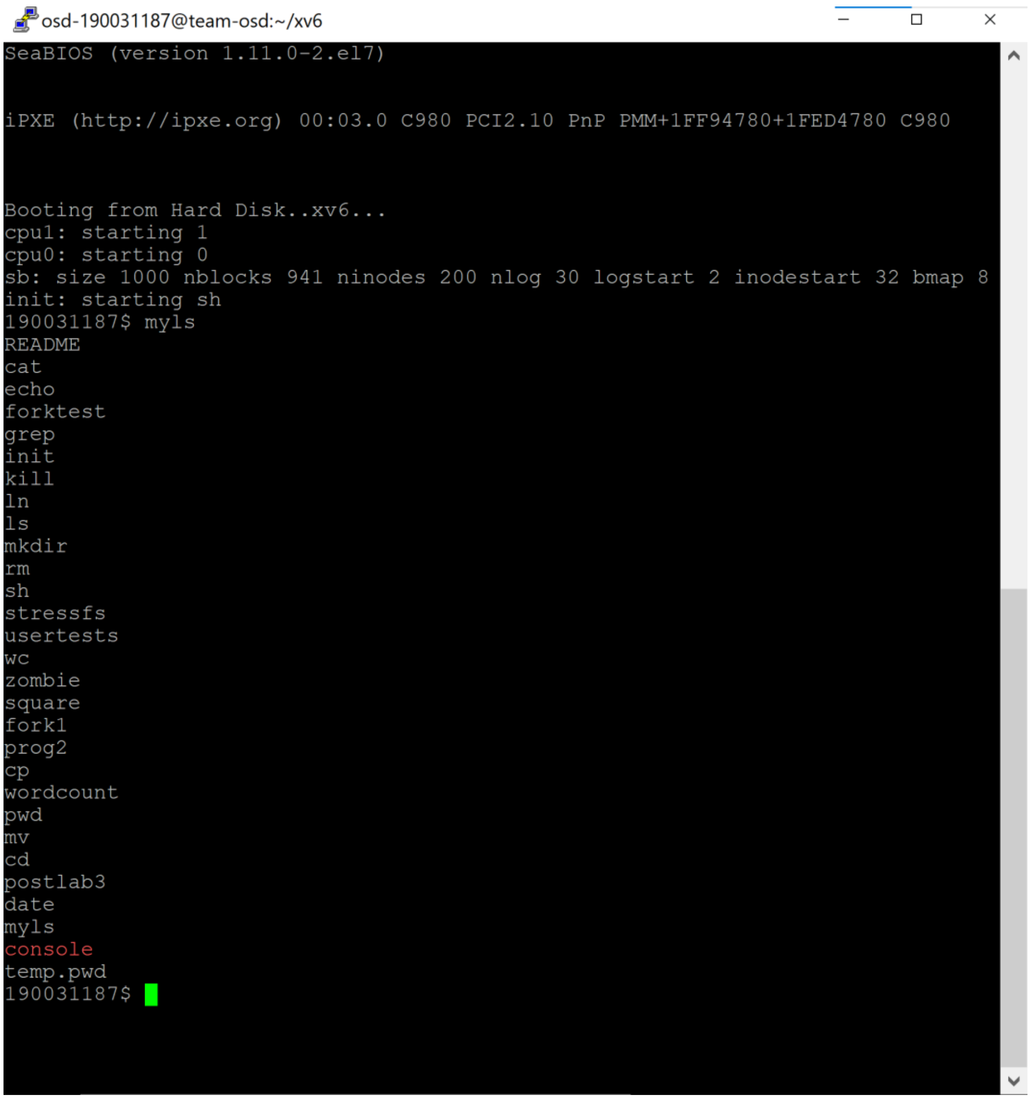
2) ls, date, head (xv6 customization (xv6 source code, https://github.com))

```
#include "types.h"
#include "stat.h"
#include "user.h"
#include "fs.h"
int lo=0; int dot=0;  int help=0;
char* fmtname(char *path)
{
  static char buf[DIRSIZ+1];
  char *p;
  // Find first character after last slash.
  for(p=path+strlen(path); p >= path && *p != '/'; p--);
  p++;
  // Return blank-padded name.
  if(strlen(p) >= DIRSIZ)
      return p;
  memmove(buf, p, strlen(p));
  memset(buf+strlen(p), ' ', DIRSIZ-strlen(p));
  return buf;
}
Void ls(char *path)
{
  char buf[512], *p;
  int fd;
  struct dirent de;
  struct stat st;
  if((fd = open(path, 0)) < 0) {
    printf(2, "ls: cannot open %s\n", path);
    return;
  }
  if(fstat(fd, &st) < 0){
    printf(2, "ls: cannot stat %s\n", path);
    close(fd);
    return;
  }
  switch(st.type){
  case T_FILE:
    printf(1, "%s %d %d %d\n", fmtname(path), st.type, st.ino, st.size);
    break;
  case T_DIR:
    if(strlen(path) + 1 + DIRSIZ + 1 > sizeof buf){
      printf(1, "ls: path too long\n");
      break;
    }
    strcpy(buf, path);
    p = buf+strlen(buf);
    *p++ = '/';
    while(read(fd, &de, sizeof(de)) == sizeof(de)){
      if(de.inum == 0)
        continue;
      if(dot==0 && de.name[0]=='.')
```

```
        continue;
      memmove(p, de.name, DIRSIZ);
      p[DIRSIZ] = 0;
      if(stat(buf, &st) < 0){
        printf(1, "ls: cannot stat %s\n", buf);
        continue;
      }
      if(lo==1)
      {
        if(st.type==T_DIR) printf(1, "\033[1m\x1B[34m%s\x1B[0m %d %d %d\n", fmtname(buf), st.type,
st.ino, st.size);
        else if(st.type==T_DEV) printf(1, "\033[1m\x1B[31m%s\x1B[0m %d %d %d\n", fmtname(buf),
st.type, st.ino, st.size);
        else printf(1, "%s %d %d %d\n", fmtname(buf), st.type, st.ino, st.size);
      }
      else
      {
        if(st.type==T_DIR) printf(1, "\033[1m\x1B[34m%s\x1B[0m\n", fmtname(buf));
        else if(st.type==T_DEV) printf(1, "\033[1m\x1B[31m%s\x1B[0m\n", fmtname(buf));
        else printf(1, "%s\n", fmtname(buf));
      }
    }
    break;
  }
  close(fd);
}

int main(int argc, char *argv[])
{
  int i;
  for(i=1;i<argc;i++)
  {
    if(argv[i][0]=='-')
    {
      if(strcmp(argv[i],"-l")==0) lo=1;
      else if(strcmp(argv[i],"-a")==0) dot=1;
      else if(strcmp(argv[i],"--help")==0) help=1;
      else printf(1,"invalid OPTIONS try 'ls --help' for more information ");
    }
  }
  if(help)
  {
    printf(1,"Usage : \033[1mls\x1B[0m [OPTION]... [FILE]...\n");
    printf(1,"List information about the FILEs (the current directory by default).\n");
    printf(1,"OPTION:\n");
    printf(1,"\t\033[1m-a\x1B[0m do not ignore entries starting with .\n");
    printf(1,"\t\033[1m-l\x1B[0m use a long listing format\n");
  }
  if(argc < 2)
  {
    ls(".");
```

```
   exit();
  }
  else if((lo==1 || dot==1 || help==1) && argc < 3)
  {
   ls(".");
   exit();
  }
  else if((lo==1 || dot==1 || help==1) && argc < 4)
  {
   ls(argv[2]);
   exit();
  }
  else
  {
   ls(argv[1]);
   exit();
  }
  exit();
}
```

OUTPUT



osd-190031187@team-osd:~/xv6

```
SeaBIOS (version 1.11.0-2.el7)


iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980



Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap 8
init: starting sh
190031187$ myls
README
cat
echo
forktest
grep
init
kill
ln
ls
mkdir
rm
sh
stressfs
usertests
wc
zombie
square
fork1
prog2
cp
wordcount
pwd
mv
cd
postlab3
date
myls
console
temp.pwd
190031187$
```

date.c

```
#include "types.h"  // this file has all the datatypes
#include "user.h"   // this file has the prototypes of all the system calls
#include "date.h"  // this file contains definition of struct rtcdate

// prototypes of all the functions used
long long power(int,int);
int check_leap(int);
void month_name(int);
void day_name(int,int,int);
void time(void);
void yesterday(void);
void today(void);
void tomorrow(void);
void particular_day(char *);
void utc_day(void);
void day(char *);

// main
int main(int argc, char *argv[])
{
        // if user only types date in the command prompt
        if(argc==1)
                today();  // this function prints today's date and current time (IST format)
        else
        {
                // if user uses -d option with the date command
                if((argc==3)&&(strcmp("-d",*(argv+1))==0))
                        day(*(argv+2));  // this function checks which option is chosen by user
                                         // it then calls a suitable function to implement that
option

                // if user uses -u option
                else if((argc==2)&&(strcmp("-u",*(argv+1))==0))
                        utc_day();  // this function prints today's date and current time (UTC
format)

                // if the user types an invalid command
                else
                        printf(1,"Invalid command. Please try again.\n");
        }
exit();
}

// this function calculate a power b
long long power(int x,int y)
```

```
{
        long long res = 1;
     int i;
        for(i=0;i<y;i++)
        {
                res = res * x;
        }
        return(res);
}

// this function whether the current year is a leap year
int check_leap(int x)
{
        int flag = 0;
        if(x%400==0)
                flag = 1;
        else if(x%100==0)
                flag = 0;
        else if(x%4==0)
                flag = 1;
        else
                flag = 0;
        return(flag);
}

// this function prints the name of the month of the year
void month_name(int x)
{
        switch(x)
        {
                case 1:printf(1," Jan");
                    break;
                case 2:printf(1," Feb");
                    break;
                case 3:printf(1," Mar");
                    break;
                case 4:printf(1," Apr");
                    break;
                case 5:printf(1," May");
                    break;
                case 6:printf(1," Jun");
                    break;
                case 7:printf(1," Jul");
                    break;
                case 8:printf(1," Aug");
                    break;
                case 9:printf(1," Sep");
                break;
            case 10:printf(1," Oct");
                break;
            case 11:printf(1," Nov");
                break;
            case 12:printf(1," Dec");
                break;
        }
}

// this function prints the name of the day of the week
void day_name(int x,int y,int z)
{
        int initial_day = 4;
        int count = 0;
     int i;
        if(x>1970)
        {
            for(i=1970;i<x;i++)
            {
                if(check_leap(i))
                    count += 366;
                else
                    count += 365;
            }
        }
        for(i=1;i<y;i++)
        {
            if(i==2)
            {
                if(check_leap(x))
                    count += 29;
                else
                    count += 28;
            }
            else if((i<8)&&(i%2==1))
                count += 31;
            else if((i<8)&&(i%2==0))
                count += 30;
            else if((i>=8)&&(i%2==0))
                count += 31;
            else
                count += 30;
        }
        int final_day = (initial_day+count+z-1)%7;
        switch(final_day)
        {
            case 0:printf(1,"Sun");
                  break;
            case 1:printf(1,"Mon");
                  break;
            case 2:printf(1,"Tue");
                  break;
            case 3:printf(1,"Wed");
                  break;
            case 4:printf(1,"Thur");
                  break;
            case 5:printf(1,"Fri");
                  break;
            case 6:printf(1,"Sat");
                  break;
       }
}

// this function prints the current time in IST format
void time()
{
      struct rtcdate r;
      if (date(&r))
      {
            printf(2, "date failed\n");
            exit();
      }
      if(r.minute+30>59)
      {
            r.hour += 6;
            r.minute = r.minute+30-59;
      }
      else
      {
            r.hour += 5;
            r.minute += 30;
      }
      if(r.hour>=24)
            r.hour -= 24;
      printf(1," %d:%d:%d",r.hour,r.minute,r.second);
}

void yesterday()
{
      struct rtcdate r;
      if (date(&r))
      {
            printf(2, "date failed\n");
```

```
		exit();
	}

	// if month is march
	if(r.month == 3)
	{
		if(check_leap(r.year))
		{
			if(r.day==1)
			{
				r.month -= 1;
				r.day = 29;
			}
			else
				r.day -= 1;
		}
		else
		{
			if(r.day==1)
			{
				r.month -= 1;
				r.day = 28;
			}
			else
				r.day -= 1;
		}
	}

	// if date is 1st Jan
	else if((r.day==1)&&(r.month==1))
	{
		r.month = 12;
		r.day = 31;
		r.year -= 1;
	}

	else
	{
		if(r.month<9)
		{
			if(r.month%2==0)
			{
				if(r.day==1)
				{
					r.month -= 1;
					r.day = 31;
				}
```

```
                else
                        r.day -= 1;
            }
            else
            {
                if(r.day==1)
                {
                        r.month -= 1;
                        r.day = 30;
                }
                else
                        r.day -= 1;
            }
        }
        else
        {
            if(r.month%2==1)
            {
                if(r.day==1)
                {
                        r.month -= 1;
                        r.day = 31;
                }
                else
                        r.day -= 1;
            }
            else
            {
                if(r.day==1)
                {
                        r.month -= 1;
                        r.day = 30;
                }
                else
                        r.day -= 1;
            }
        }
    }
    day_name(r.year,r.month,r.day);  // prints the name of yesterday's day of the week
    month_name(r.month);  // prints the name of the yesterday's month of the year
    printf(1," %d",r.day);  // prints yesterday's date
    time();  // prints the current time (IST format)
    printf(1," IST");
    printf(1," %d\n",r.year);  // prints yesterday's year
}

// this function prints today's date and current time (IST format)
```

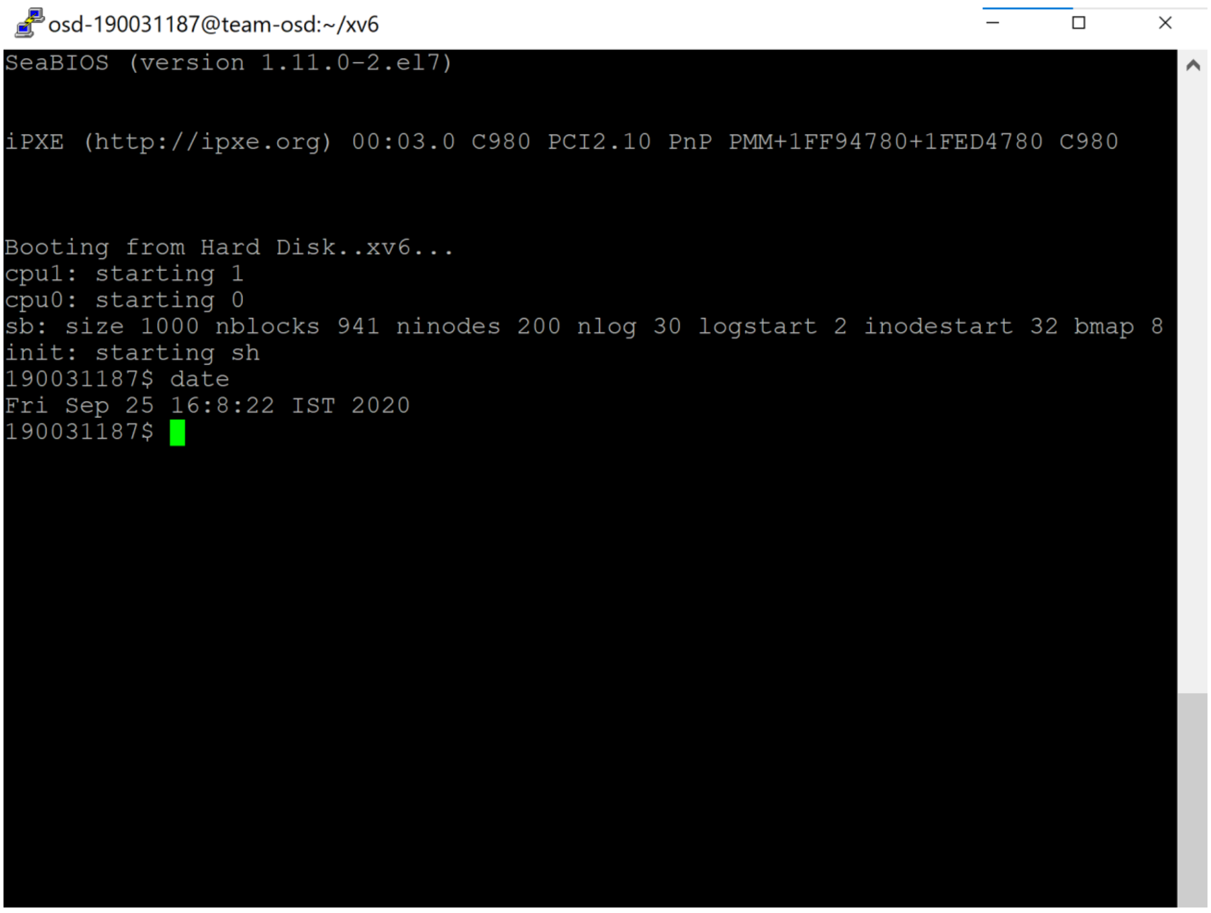
```
void today()
{
        struct rtcdate r;
        if (date(&r))
        {
                printf(2, "date failed\n");
                exit();
        }
        day_name(r.year,r.month,r.day);       // prints the name of day of the week
        month_name(r.month);  // prints the name of the month of the year
        printf(1," %d",r.day);  // prints the today's date
        time();  // prints the current time (IST format)
        printf(1," IST");
        printf(1," %d\n",r.year);  // prints the current year
}

// this function prints tomorrow's date and time (IST format)
void tomorrow()
{
        struct rtcdate r;
        if (date(&r))
        {
                printf(2, "date failed\n");
                exit();
        }

        // if month is Feb
        if(r.month == 2)
        {
                if(check_leap(r.year))
                {
                        if(r.day==29)
                        {
                                r.month += 1;
                                r.day = 1;
                        }
                        else
                                r.day += 1;
                }
                else
                {
                        if(r.day==28)
                        {
                                r.month += 1;
                                r.day = 1;
                        }
                        else
```

```
					r.day += 1;
			}
		}

		// if the date is 31st Dec
		else if((r.day==31)&&(r.month==12))
		{
			r.month = 1;
			r.day = 1;
			r.year += 1;
		}

		else
		{
			if(r.month<8)
			{
				if(r.month%2==1)
				{
					if(r.day==31)
					{
						r.month += 1;
						r.day = 1;
					}
					else
						r.day += 1;
				}
				else
				{
					if(r.day==30)
					{
						r.month += 1;
						r.day = 1;
					}
					else
						r.day += 1;
				}
			}
			else
			{
				if(r.month%2==1)
				{
					if(r.day==30)
					{
						r.month += 1;
						r.day = 1;
					}
					else
```

```
                                        r.day += 1;
                        }
                        else
                        {
                                if(r.day==31)
                                {
                                        r.month += 1;
                                        r.day = 1;
                                }
                                else
                                        r.day += 1;
                        }
                }
        }
        day_name(r.year,r.month,r.day);  // prints the name of tomorrow's day of the week
        month_name(r.month);  // prints the name of the tomorrow's month of the year
        printf(1," %d",r.day);  // prints tomorrow's date
        time();  // prints the current time (IST format)
        printf(1," IST");
        printf(1," %d\n",r.year);  // prints tomorrow's year
}

// if user uses -d option with a particular date
void particular_day(char *x)
{
        int flag = 1;
     int i;
        for(i=0;i<4;i++)
        {
                if(*(x+i)=='-')
                {
                        flag = 0;
                        break;
                }
        }
        for(i=5;i<7;i++)
        {
                if(*(x+i)=='-')
                {
                        flag = 0;
                        break;
                }
        }
        for( i=8;i<10;i++)
        {
                if(*(x+i)=='-')
                {
                        flag = 0;
                        break;
                }
        }
        if(flag==0)
        {
                printf(1,"date: invalid date %s\n",x);
                exit();
        }
        int y=0;
        int m=0;
        int d=0;
        for( i=0;i<4;i++)
        {
                y += (*(x+i) - '0')*power(10,3-i);
        }
        for( i=5;i<7;i++)
        {
                m += (*(x+i) - '0')*power(10,6-i);
        }
        for( i=8;i<10;i++)
        {
                d += (*(x+i) - '0')*power(10,9-i);
        }
        if((m>12)||(d>31)||(m<1)||(d<1)||(y<1970))
        {
                printf(1,"date: invalid date %s\n",x);
                exit();
        }
        else if(m==2)
        {
                if(d>28)
                {
                        if(check_leap(y))
                        {
                                if(d>29)
                                {
                                        printf(1,"date: invalid date %s\n",x);
                                        exit();
                                }
                        }
                        else
                        {
                                printf(1,"date: invalid date %s\n",x);
                                        exit();
                        }
                }
```

```
	}
	day_name(y,m,d);
	month_name(m);
	printf(1," %d",d);
	printf(1," 00:00:00 IST");
	printf(1," %d\n",y);
}

// this function prints today's date and current time (UTC format)
void utc_day()
{
	struct rtcdate r;
	if (date(&r))
	{
		printf(2, "date failed\n");
		exit();
	}
	day_name(r.year,r.month,r.day);	// prints the name of today's day of the week
	month_name(r.month);  // prints the name of the today's month of the year
	printf(1," %d",r.day);  // prints today's date
	printf(1," %d:%d:%d",r.hour,r.minute,r.second);  // prints the current time (UTC
format)
	printf(1," UTC");
	printf(1," %d\n",r.year);  // prints current year
}

// this function checks which option is chosen by user
// it then calls a suitable function to implement that option
void day(char *x)
{
	switch(*(x+2))
	{
		case 'd':
		case 'w':today();
		      break;
		case 'm':tomorrow();
		      break;
		case 's':yesterday();
		      break;
		default:particular_day(x);
			  break;
	}
}
```

OUTPUT



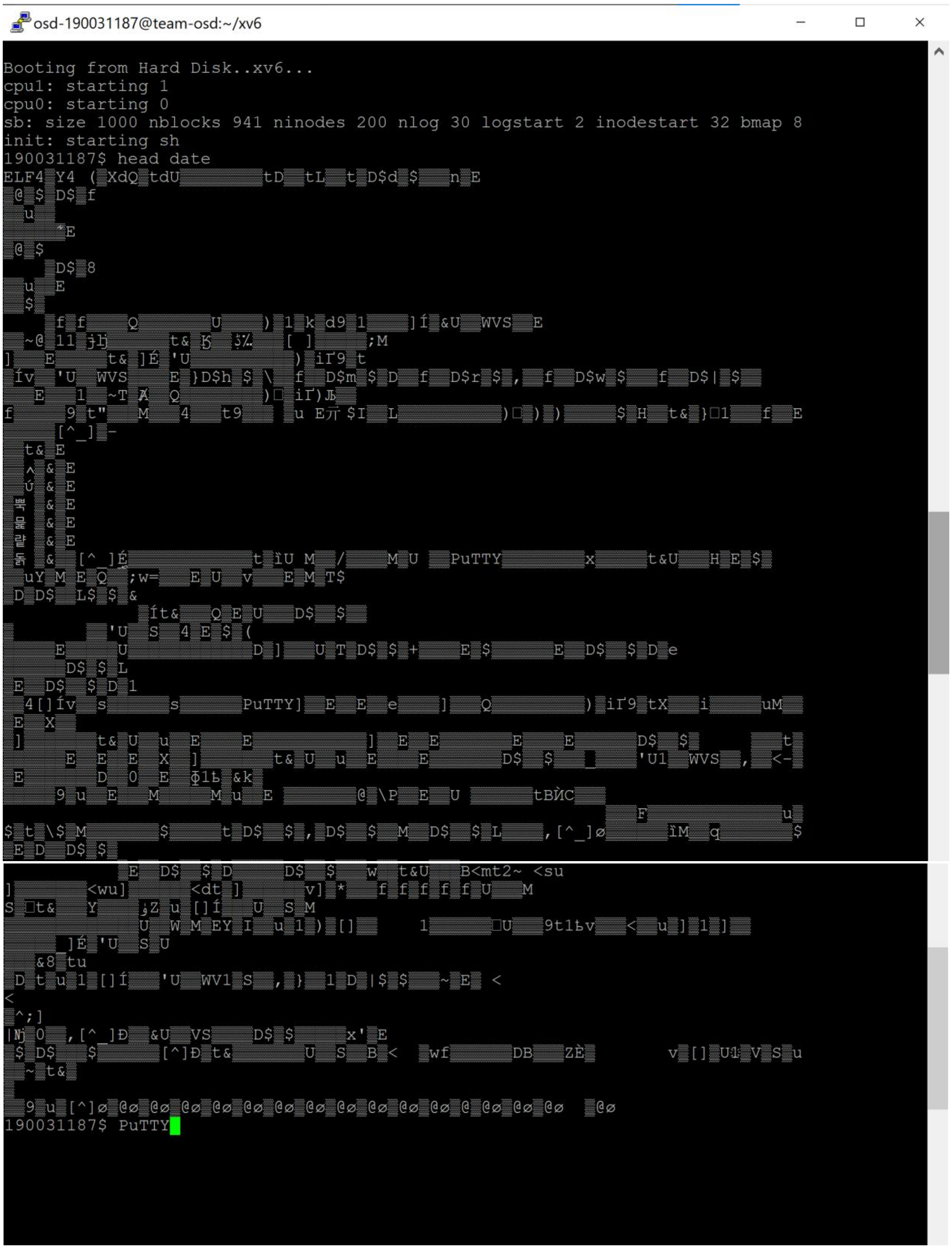
osd-190031187@team-osd:~/xv6

```
SeaBIOS (version 1.11.0-2.el7)


iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980



Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap 8
init: starting sh
190031187$ date
Fri Sep 25 16:8:22 IST 2020
190031187$
```

head.c

```
#include "types.h"
#include "stat.h"
#include "user.h"
char buf[512];
void head(int fd, char *name, int line)
{
int i, n; //here the size of the read chunk is defined by n, and i is used to keep a track of the
chunk index
int l, c; //here line number is defined by l,and the character count in the string is defined by
c
l = c = 0;
while((n = read(fd, buf, sizeof(buf))) > 0 ){
   for(i=0;i<=n ;i++){                              //print the characters in the line
     if(buf[i]!='\n'){
            printf(1,"%c",buf[i]);
     }
                          //if the number of lines is equal to l, then exit
     else if (l == (line-1)){
            printf(1,"\n");
            exit();
     }
//if the number of lines is not equal to l, then jump to next line and increment the value of l
     else{
          printf(1,"\n");
          l++;
     }
   }
  }
  if(n < 0){
   printf(1, "head: read error\n");
   exit();
  }
}
int main(int argc, char *argv[]) {
     int i;
     int fd = 0;          // when the file is not specified, then it will take input from the user
     int x = 10;          // will read the first 10 lines by default
     char *file;          // pointer to the name of the file
     char a;
     file = ""; // in the case when no file name is specified, it will take input from the user
     if (argc <= 1) {
          head(0, "", 10);           // handles the default case of taking input from user for 10 lines
          exit();
     }
     else {
          for (i = 1; i < argc; i++) {
                    a = *argv[i];           // assigns the char value of the argv to the var a

                if (a == '-') { // it means that -NUM is provided, hence limited number of lines are
to be printed
                              argv[i]++;
                     x = atoi(argv[i]++);
                }
                else {      // if a !='-' then it implies that number of lines are not defined and hence
default lines will print
               if ((fd = open(argv[i], 0)) < 0) {// this will execute if the file is unable to open
                     printf(1, "head: cannot open %s\n", argv[i]);
                     exit();
                     }
                }
          }
          head(fd, file, x);
          close(fd);
          exit();
     }
}
```

OUTPUT



```
Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap 8
init: starting sh
190031187$ head date
ELF4▒Y4 (▒XdQ▒tdU▒▒▒▒▒▒▒▒tD▒▒tL▒▒t▒D$d▒$▒▒▒n▒E
▒@▒$ D$▒f
▒▒u▒▒
▒▒▒▒▒*E
▒@▒$
     ▒D$▒8
▒▒u▒▒E
▒▒$▒
    ▒f▒f▒▒▒▒Q▒▒▒▒▒▒▒▒U▒▒▒▒)▒1▒k▒d9▒1▒▒▒▒]í▒&U▒▒WVS▒▒E
▒▒~@▒11▒ǰǰ▒▒▒▒▒t&▒Ƙ▒▒ʒ⅞▒▒▒[ ]▒▒▒▒;M
]▒▒E▒▒▒▒▒t&▒]É▒'U▒▒▒▒▒▒▒▒▒▒)▒iΓ9▒t
Ív▒▒'U▒▒WVS▒▒▒▒E▒}D$h▒$▒\▒▒f▒▒D$m▒$▒D▒▒f▒▒D$r▒$▒,▒▒f▒▒D$w▒$▒▒▒f▒▒D$|▒$▒▒
▒▒E▒▒▒1▒▒~T▒Ă▒Q▒▒▒▒▒▒▒▒▒)□▒iΓ)Љ▒▒
f▒▒▒▒▒9▒t"▒▒▒M▒▒▒4▒▒▒t9▒▒▒▒u E亓$I▒▒L▒▒▒▒▒▒▒▒▒▒▒)□▒)▒)▒▒▒▒▒$▒H▒▒t&▒}□1▒▒▒f▒▒E
▒▒▒▒▒[^_]▒-
▒t&▒E
▒^▒&▒E
▒ú▒&▒E
▒뽁▒&▒E
▒뭏▒&▒E
▒랕▒&▒E
▒됵▒&▒▒[^_]É▒▒▒▒▒▒▒▒▒▒▒t▒ìU▒M▒▒/▒▒▒▒M▒U▒▒PuTTY▒▒▒▒▒▒▒▒x▒▒▒▒▒t&U▒▒▒H▒E▒$▒
▒uY▒M▒E▒Q▒▒;w=▒▒▒E▒U▒▒v▒▒▒E▒M▒T$
▒D▒D$▒▒L$▒$▒&
             ▒ít&▒▒▒Q▒E▒U▒▒▒D$▒▒$▒▒
▒        ▒▒'U▒▒S▒▒4▒E▒$▒(
▒▒▒▒▒E▒▒▒▒U▒▒▒▒▒▒▒▒▒▒▒D▒]▒▒▒U▒T▒D$▒$▒+▒▒▒▒E▒$▒▒▒▒▒▒▒E▒▒D$▒▒$▒D▒e
▒▒▒▒▒▒D$▒$▒L
▒E▒▒D$▒▒▒$▒D▒1
▒▒4[]Ív▒▒s▒▒▒▒▒▒s▒▒▒▒▒▒PuTTY]▒▒E▒▒E▒▒e▒▒▒▒]▒▒▒Q▒▒▒▒▒▒▒▒▒)▒iΓ9▒tX▒▒▒i▒▒▒▒▒uM▒▒
▒E▒▒X▒▒
▒]▒▒▒▒▒▒t&▒U▒▒u▒▒E▒▒▒E▒▒▒▒▒▒▒▒▒▒▒▒]▒▒E▒▒E▒▒▒▒▒▒▒E▒▒▒▒E▒▒▒▒▒D$▒▒$▒▒▒▒▒▒▒t▒
▒▒▒▒▒▒E▒▒E▒▒E▒▒X▒▒]▒▒▒▒▒▒▒▒t&▒U▒▒u▒▒E▒▒▒▒E▒▒▒▒▒▒▒▒D$▒▒▒$▒▒▒▒_▒▒▒'U1▒▒WVS▒▒,▒▒<-▒
▒E▒▒▒▒▒▒D▒▒0▒▒E▒▒ф1Ь▒&k
▒▒▒▒▒9▒u▒▒E▒▒▒M▒▒▒▒▒M▒u▒▒E▒▒▒▒▒▒▒▒▒@▒\P▒▒E▒▒U▒▒▒▒▒▒▒▒tВЙC▒▒▒
                                                      ▒▒▒F▒▒▒▒▒▒▒▒▒▒▒▒▒▒u▒
$▒t▒\$▒M▒▒▒▒▒▒▒▒$▒▒▒▒▒▒t▒D$▒▒$▒,▒D$▒▒$▒▒M▒▒D$▒▒$▒L▒▒▒▒,[^_]ø▒▒▒▒▒▒▒ĩM▒▒q▒▒▒▒▒▒▒▒$
▒E▒D▒▒D$▒$▒
           ▒E▒D$▒▒$▒D▒▒▒▒▒D$▒▒$▒▒▒▒w▒▒t&U▒▒▒B<mt2~ <su
]▒▒▒▒▒▒▒▒<wu]▒▒▒▒▒▒<dt▒]▒▒▒▒▒▒▒v]▒*▒▒▒f▒f▒f▒f▒f▒U▒▒▒M
S▒□t&▒▒▒Y▒▒▒▒¸Z▒u▒[]í▒▒▒U▒▒S▒M
▒▒▒▒▒▒▒▒▒▒▒▒▒U▒▒W▒M▒EY▒I▒▒u▒1▒)▒[]▒▒    1▒▒▒▒▒▒□U▒▒▒9t1Ьv▒▒▒<▒▒u▒]▒1▒]▒▒
▒▒▒▒▒▒]É▒'U▒▒S▒U
▒▒▒▒&8▒tu
▒D▒t▒u▒1▒[]í▒▒▒'U▒▒WV1▒S▒▒,▒}▒▒▒1▒D▒|$▒$▒▒▒▒~▒E▒ <
<
▒^;]
|Ǌ▒0▒▒,[^_]Đ▒▒&U▒▒VS▒▒▒▒D$▒$▒▒▒▒▒▒x'▒E
▒$▒D$▒▒▒$▒▒▒▒▒▒[^]Đ▒t&▒▒▒▒▒▒▒▒U▒▒S▒▒B▒<▒▒▒wf▒▒▒▒▒▒DB▒▒▒ZÈ▒       v▒[]▒U1▒V▒S▒u
▒▒~▒t&▒
▒
▒▒9▒u▒[^]ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@ø▒@▒@ø▒@ø▒@ø  ▒@ø
190031187$ PuTTY
```

Out is the ascii values of the original file . date is the file name